Data Science in R (DSR) – Testat 1

Markus Gumbel

17. Oktober 2023

Inhaltsverzeichnis

Hinweise

Aktion		Datum	Ablauf
Ausgabe	Do.,	19.10.23	über Moodle
Abgabe	Fr.,	10.11.23, 14:00 Uhr	über Moodle
Testat	Mo.,	13.11.23	als Präsentation

- Die Abnahme der Testate gilt als Prüfungsleistung. Bei einer Verhinderung durch Krankheit ist eine ärztliche Bescheinigung der Arbeitsunfähigkeit vorzulegen.
- Geben Sie Ihre Lösungen fristgerecht via Moodle ab. Es werden dort folgende Abgaben erwartet (Ausarbeitungen in einem anderen Format werden nicht berücksichtigt):
  - Für jede Aufgabe ein R-Notebook (als *.Rmd-Datei) und etwaige benötigten R-Skripte (*.R).
  - Genau ein PDF mit allen Lösungen, das außerdem die Gruppennummer und alle Namen enthält.
- Nutzen Sie das R-Notebook, das es für jede Aufgabe gibt und **fügen Sie dort Ihre Lösungen ein**, d.h. Source-Code und normaler Text.
- **Kommentieren Sie Ihre Lösungen ausführlich** – entweder als normalen Text im Notebook oder als Kommentar in R-Anweisungen. Ohne Kommentare gibt es Punktabzug.
- Jedes abgegebene R-Notebook muss ohne Fehler auf einem anderen Rechner mit R (4.1 oder höher) ausgeführt werden können und ein HTML-Dokument erzeugen.

Installationsanweisungen für die benutzen Packages sind nicht nötig – laden Sie aber die Packages im Notebook.

- Während der Abnahme sind die Ergebnisse am Rechner live zu demonstrieren.
- Bei der Abnahme der Übung ist der Studentenausweis vorzulegen.

1 Eigenes Histogramm

Es soll ein eigenes Histogramm erzeugt werden. Der Dateiname für das Skript ist `myhistogram.R`.

1.1 Funktion `myhistogram`

Programmieren Sie in R die Funktion `myhistogram`, die als Parameter `x` einen Vektor aus Zahlen erhält. Die Zahlen werden in `n` Intervalle einsortiert, und es wird gezählt, wie oft eine Zahl in einem Intervall vorkommt. Der Rückgabewert ist eine Liste mit den Einträgen `borders`, die die $n+1$ Intervallgrenzen enthalten und `counts`, der die Anzahlen enthält.

- Die n Intervalle sollen gleich groß sein (Δb), d.h. für die Intervallgrenzen $b_1, b_2, \ldots, b_{n+1}$ gilt $\frac{b_{i+1}-b_i}{n} = \Delta b$ für $i = 1, 2, \ldots n$.
- Die äußeren Grenzen b_1 und b_n sollen als optionale Parametern `min` und `max` an die Funktion übergeben werden. Werte aus `x`, die zu keinem Intervall gehören, sollen ignoriert werden. Es wird aber eine Warnung ausgegeben, die sagt, welche Zahlen außerhalb des Bereichs liegen.
- Eine Zahl z gehört zum i-ten Intervall, falls $b_i \leq z < b_{i+1}$ gilt.

Bis auf `x` sollen alle Parameter optional sein. Überlegen Sie sinnvolle Default-Werte.

Es ist natürlich **nicht** erlaubt, in der eigenen Funktion andere Funktionen zu nutzen, die ein Histogramm erzeugen.

Hier ein Beispiel:

```
# Bitte mit echter Lösung in myhistogram.R ersetzen:
myhistogram = function(x) {
  list(borders = c(), counts = c())
}
x = seq(-5, 6, by = 1 / 3)
l = myhistogram(x)
print(l$borders)
print(l$counts)
```

1.2 Beispieldaten

Hier zunächst zwei Beispiele.

1.2.1 Beispiel 1

Es wird eine Warnung ausgeben:

```
## Warning in myhistogram(x, n = 10, min = -5, max = 6): Zahl(en) außerhalb
## Intervallgrenzen: 6
```

```
x = seq(-5, 6, by = 1 / 3)
# myhistogram(x, n = 10, min = -5, max = 6)
solution = list(
  borders = c(-5.0, -3.9, -2.8, -1.7, -0.6, 0.5, 1.6, 2.7, 3.8, 4.9, 6.0),
  counts = c(4, 3, 3, 4, 3, 3, 4, 3, 3, 3)
)
```

1.2.2 Beispiel 2

```
x = seq(-5, 6, by = 1 / 3)
# myhistogram(x, n = 5, min = -10, max = 10)
solution = list(borders = c(-10, -6, -2, 2, 6, 10),
                counts = c(0, 9, 12, 12, 1))
```

1.2.3 Beispiel 3

Testen Sie nun hier Ihre Funktion mit weiteren Datensätzen.

1.2.4 Beispiel 4

Testen Sie nun hier Ihre Funktion mit weiteren Datensätzen.

1.2.5 Beispiel 5

Testen Sie nun hier Ihre Funktion mit weiteren Datensätzen.

1.3 Barplot

Nutzen Sie Ihre Funktion `myhistogram` und erzeugen Sie einen Barplot mit *ggplot*. Die x-Achse zeigt dabei die Mitte des Intervalls und die y-Achse die Anzahl der Elemente in dieser Klasse.

Tipp: Der Parameter `stat` von `geom_bar` ist wichtig.

Vervollständigen Sie den Chunk. Die Kommentare sollen zu Anweisungen umgewandelt werden:

```
set.seed(1)
x = rnorm(0, 1, n = 1000)
# h = myhistogram(x, n = 20)
# ggplot() + ...
```

2 Visualisierung von Datensätzen

In diesem Abschnitt sollen alle Graphiken mit *ggplot* und alle Tabellen mit `kable` erstellt werden.

2.1 Körpergewicht und Gehirngewicht bei Säugetieren

Nutzen Sie den Datensatz `MASS::mammals`. In der Hilfe finden Sie Hinweise, was dort gezeigt ist.

2.1.1 Körpergewicht vs. Gehirngewicht

Erzeugen Sie diese Graphik, indem Sie den nachfolgenden Chunk vervollständigen. Die gezeigten Tiernamen sind Pig, Rat, African elephant, Chimpanzee, Cat, Human, Little brown bat.

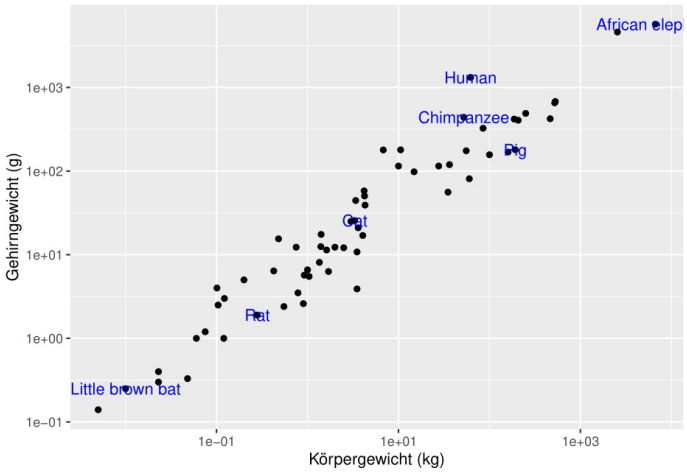

Abbildung 1: Körper- vs. Gehirngewicht.

Tipp: Sie dürfen (und sollen) weitere Libraries nutzen, wenn diese hilfreich sind.

```
# Ihre Lösung:
```

2.1.2 Gehirn- zu Körpergewicht-Verhältnis

Geben Sie diejenigen 10 Tiere als Tabelle im Notebook aus, die das größte Gehirn- zu Körpergewicht-Verhältnis r haben. Die Liste soll nach r absteigend sortiert sein und den Tiernamen und r enthalten.

Vervollständigen Sie diesen Chunk:

```
# Ihre Lösung:
```

Geben Sie nun – wie eben – diejenigen 10 Tiere als Tabelle aus, die das **kleinste** Gehirn- zu Körpergewicht-Verhältnis r haben. Die Liste soll nach r absteigend sortiert sein.

Vervollständigen Sie diesen Chunk:

```
# Ihre Lösung:
```

2.2 Blutdruckveränderung bei Medikamentengabe im Tierversuch

Nutzen Sie den Datensatz `MASS::Rabbit`. In der Hilfe finden Sie Hinweise, was dort gezeigt ist.

2.2.1 Überblick über Verlauf bei allen Kaninchen

Plotten Sie im folgenden Chunk den Verlauf der Blutdruckveränderung (y-Achse) bei gegebener Dosis Phenylbiguanide (x-Achse). Dies soll in einem Diagramm mit Unterdiagrammen erfolgen: ein Unterdiagramm zeigt den Verlauf für je ein Kaninchen und der Behandlung (Placebo oder MDL 72222).

```
# Ihre Lösung:
```

2.2.2 Boxplots der Blutdruckänderung je Dosis

Erzeugen Sie ein Diagramm, das in zwei Unterdiagrammen für die Placebo- und die MLD-Gruppe Boxplots erstellt. Die Boxplots geben die Verteilung der Blutdruckänderung je Dosis an. In Anlehnung an das obige Diagramm sollen die Boxplots vertikal ausgerichtet sein.

```
# Ihre Lösung:
```

3 Covid19: Impffortschritt in Deutschland

In dieser Aufgabe geht es um den Verlauf der Corona-Impfungen in Deutschland. Die folgenden URLs enthalten Daten ab 2020:

- https://impfdashboard.de/static/data/germany_vaccinations_timeseries_v2.tsv
- https://impfdashboard.de/static/data/germany_deliveries_timeseries_v2.tsv
- https://impfdashboard.de/static/data/germany_vaccinations_by_state.tsv

Sie sind der Webseite https://impfdashboard.de entnommen.

3.1 Einlesen der Daten

Lesen Sie die drei Dateien je in einen Data Frame ein mit den Variablennamen:

- `vacc`
- `deliv`
- `vaccState`

Wandeln Sie die Datums- und Zeitangaben von einem String in ein R-Datumsobjekt um. Geben Sie die ersten drei Zeilen und Spalten dieser Data Frames aus.

```
# Ihre Lösung:
# germany_vaccinations_timeseries_v2.tsv in Variable vacc
# germany_deliveries_timeseries_v2.tsv in Variable deliv
# germany_vaccinations_by_state.tsv in Variable vaccState
```

3.2 Verimpfte Impfdosen pro Tag

Es soll untersucht werden, wie oft welcher Impfstoff an welchem Tag verimpft wurde.

3.2.1 Transformation

Der Data Frame `vacc` enthält leider keine Angaben, wie oft ein Impfstoff eines Herstellers täglich verabreicht wurde. Erzeugen Sie aus `vacc` einen neuen Data Frame `vacc2`, der die folgende Struktur hat:

Tabelle 2: Neue Struktur: Data Frame `vacc2`.

Datum	Hersteller	Impfdosen pro Tag
09.04.21	biontech	123456
09.04.21	moderna	12345
...	...	...

Wie Sie die Impfstoffe (biontech, moderna, astra) nennen, bleibt Ihnen überlassen – solange die Bezeichnungen konsistent und schlüssig sind.

Geben Sie die letzten Zeilen von `vacc2` als `kable` aus. Tipp: `tail` gibt die letzten Zeilen eines Data Frames an (analog zu `head`).

```
# Ihre Lösung:
```

3.2.2 Plot der täglichen Impfdosen nach Hersteller

Plotten Sie mit *ggplot* den Verlauf der täglichen Impfdosen für jeden Hersteller. Die x-Achse zeigt das Datum und die y-Achse die Anzahl der Impfdosen pro Tag. Überlegen Sie, welcher Diagrammtyp dafür am besten geeignet ist.

```
# Ihre Lösung:
```

3.3 Zeitverzug Auslieferung bis Verimpfung

Es soll untersucht werden, wie schnell gelieferte Impfmengen der einzelnen Impfstoffe auch verimpft wurden.

Es bietet sich dafür an, die akkumulierten Impfdosen mit den akkumulierten Impflieferungen zeitlich plotten. Je größer die Lücke zwischen der Liefermenge und der Impfungen ist, desto mehr Impfstoff blieb liegen. Die Graphik soll Angaben für ganz Deutschland und nicht für die einzelnen Bundesländer zeigen.

Hinweis: Auch hier ist eine Vorverarbeitung der Daten nötig.

Plotten Sie dies mit *ggplot*:

```
# Ihre Lösung:
```